Approach Perfect

A Field Guide to Scrum Events

Dear Fellow Agilist,

This field guide is intended to provide the “bare minimum” instruction required to successfully bootstrap the five standard Scrum Events and Backlog Refinement. It is written to illustrate both “textbook” Scrum (scrumguides.org) as well as how it is conventionally taught and practiced.

This guide doesn’t attempt to account for all possible variations in practice, nor does it pretend to be a “best practice” or even a “good practice” in all contexts. A high-performing team will eventually find a better framework than “textbook” Scrum for their environment. However, I suggest that you deviate from it only with well-formed experiments after practicing diligently for several sprints under the guidance of a knowledgeable Scrum Master or Agile Coach.

Think of this in the same way that a wise martial arts student will long practice the basic forms under the guidance of a skilled teacher before trying to promote novel forms themselves. There will be a time for deviation—*it could even be today*—but always do so with humility and honest curiosity.

Sometimes this document repeats itself. That’s intentional as this allows you to print out just the pages you need for any given event or context and still have the information you need.

I’ll earn a small commission if you buy any of the few books or supplies I recommend through the Amazon links in this document. Thanks for your support. ❤️

I’ll keep working to improve this guide for you. You can always find the latest version of it at ap.tips/eventsfieldguide. This particular edition was last updated on April 3rd, 2020. I invite your feedback, so please contact me at https://approachperfect.com with your suggestions for improvement.

Best of luck to you in your Agile journey. May you always experience tremendous joy, satisfaction, and purpose in your collaboration with others. It is certainly possible and you inherently deserve it.

Warmly,

Chris Gagné
Approach Perfect, Limited (https://approachperfect.com)
Wellington, New Zealand

Scrum Events at a Glance

For a 16-minute video overview of Scrum—including the Scrum events—see ap.tips/scrumin16.

Event	Purpose	Participants	Timing and Usual Time-box	Maximum Time-box
Sprint	Deliver a usable, potentially-releasable "*Done*" *Increment* that accomplishes the *Sprint Goal*.	The entire *Scrum Team.* Stakeholders and other subject matter experts (SMEs) also participate daily.	1–4 weeks or monthly. Most commonly 2 weeks.	One month.
Sprint Planning ap.tips/planning	Identify a *Sprint Goal.* Plan the *Increment* by selecting *Backlog Items* and co-designing how they will be completed.	The entire *Scrum Team.* Stakeholders and other SMEs may also be invited.	Typically 1-2 hours per week of *Sprint* duration. Always at the beginning of the sprint.	8 hours for a 1-month *Sprint*.
Daily Scrum ap.tips/scrum	Inspect progress towards the *Sprint Goal* and adapt. Plan work for the next 24 hours.	The entire *Scrum Team.* Stakeholders may attend as observers.	15 minutes or less every day. Folks can linger longer for informal collaboration if they like.	15 minutes.
Sprint Review ap.tips/review	Get stakeholder feedback and improve the *Product Backlog*. Build trust and transparency.	The entire *Scrum Team,* stakeholders, and even end users should all attend.	15-60 minutes per week of *Sprint* duration. Formal end of the Sprint. 1 hour for a 2-week is ideal.	4 hours for a 1-month *Sprint*.
Sprint Retrospective ap.tips/retro	Fearlessly and collaboratively identify and plan improvement opportunities as the *Scrum Team*.	The entire *Scrum Team*. Others must not attend unless freely invited by all on the *Scrum Team*.	1–3 hours per sprint. 1½ hours strongly recommended for a 2-week *Sprint*.	3 hours for a 1-month *Sprint*.
Backlog Refinement ap.tips/refinement (not officially a Scrum event)	Get *Backlog Items* "Ready" for the next *Sprint*.	Most or all of the *Scrum Team*. Stakeholders and other SMEs as needed.	1½–4 hours per week. Newer backlogs usually need more refinement.	10% of the *Development Team's Capacity*.

The Sprint

Also known as an “Iteration” in Extreme Programming.

Goal

- Deliver a usable, potentially-releasable product *Increment* that meets the *Definition of “Done”* (ap.tips/dod) and accomplishes the *Sprint Goal*.

Timing

- 1–4 weeks. (Agile principles #1, 3, and 7; ap.tips/principles) 2-week *Sprints* are increasingly the most common.
- Each sprint starts immediately after the last ends. Don’t extend the sprint duration.

Participants

- **Required:** The self-organizing (Agile principle #11) *Scrum Team*—a *Product Owner* (ap.tips/po), *Scrum Master* (ap.tips/sm), and the cross-functional *Development Team* (ap.tips/devteam)—are the people primarily responsible for the success of the Sprint.
    - Scrum does not recognize individual titles for *Development Team* members such as software engineer, QA tester, or even designer. There are no “sub-teams” in Scrum.
- **Invited:** The *Scrum Team* should work together daily with business people, who are often invited to participate in *Sprint Planning*, *Daily Scrum*, and *Sprint Reviews*. (Agile principle #4)

Inputs

- Enough prioritized *Backlog Items* that are “*Ready*” (ap.tips/dor) to work on.

Outputs

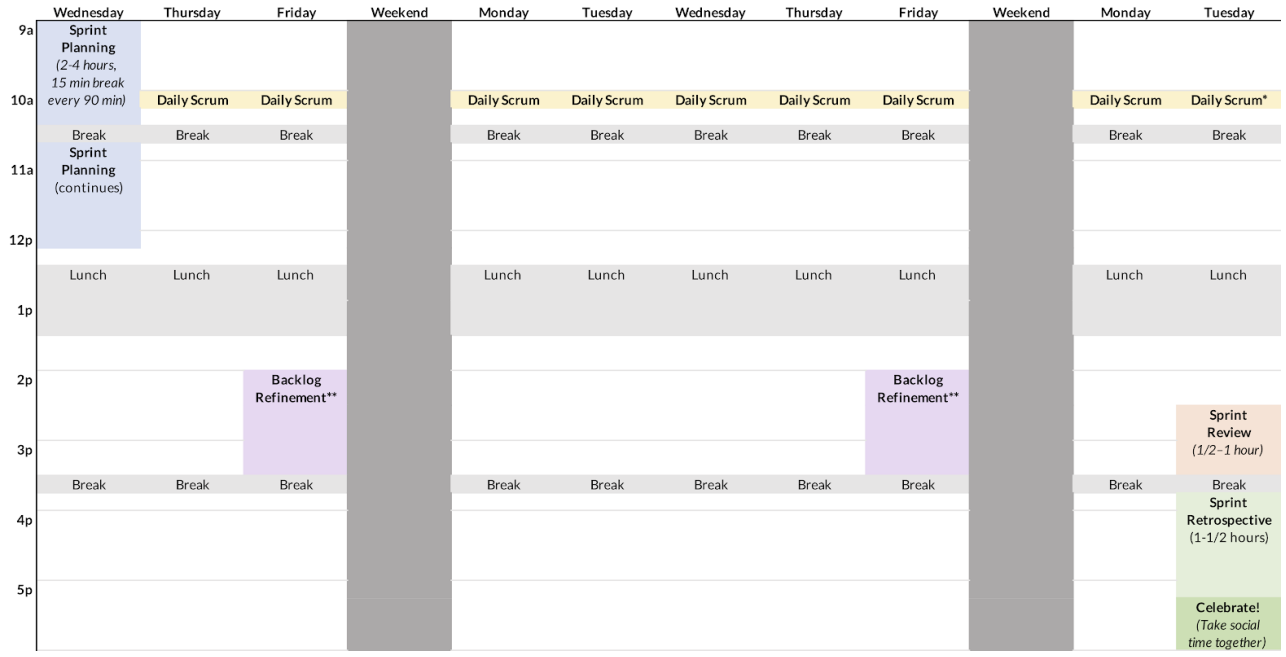
- A usable, potentially-releasable product *Increment*.
- Increased useful knowledge about product, customers, competition, and the *Scrum Team*.

Tips

- Teams newly starting with Scrum should consider 1-week sprints to “facilitate more rapid inspect/adapt cycles.” (See Jeff Sutherland et al. “Shock Therapy”, ap.tips/shocktherapy).
- It is a common misconception that a one-week sprint has twice the planning overhead. Although a team conducting one-week sprints would have two *Sprint Planning* (ap.tips/planning), *Sprint Review* (ap.tips/review), and *Sprint Retrospectives (ap.tips/retro)*, the *Planning* and *Review events* take about 40% less time for one-week sprints because there are fewer *Backlog Items* to plan and review with stakeholders and end users.
- Do not allow the team to run at an unsustainable pace. (Agile principle #8)
- Do not neglect the quality of the product. (Agile principle #9)
- The *Development Team* and *Product Owner* can clarify and change the Sprint’s scope as more is learned, but the *Sprint Goal* should be preserved whenever possible.

- The *Product Owner* may cancel a *Sprint* before the time-box has elapsed, usually in consultation with the *Scrum Master* and *Development Team*. This usually happens when the *Sprint Goal* is obsolete. A *Sprint* cancellation is traumatic to the team and consumes resources, so should be avoided if possible.

Sample Sprint Agenda (Two-Week Sprint)



A typical Scrum team uses two-week Sprints. This example schedule shows:

- A ***Sprint*** starting on a Wednesday, not a Monday as you'd might expect. Many teams start their sprints in the middle of the week to lower the risk of holidays forcing a rescheduling.
- A **3-hour *Sprint Planning***. *Sprint Planning* should take between 1–2 hours per week of *Sprint* duration. Allow a 10-15 minute break every 60-90 minutes during *Sprint Planning* and other events; our attention span just doesn't last longer than that.
- A **15-minute *Daily Scrum*** every day at the same time. *Some teams will cancel the *Daily Scrum* on the last day of the sprint, but we recommend against that as the team could use this time to coordinate their final effort against the *Sprint Goal*.
- Up to **10%** of the *Sprint* is spent conducting ***Backlog Refinement***. †*This is not an official Scrum event*, it's just an activity that happens through the sprint. However, most teams regularly schedule *Backlog Refinement* events to block off time and keep a good cadence.
- A **1-hour Sprint Review**. This gives the team enough time to get meaningful feedback from stakeholders. Less mature teams try to complete this in 30 minutes, but we've found that this becomes a rapid-fire, one-sided demo, not a true inspection and adaptation opportunity.
- A **1½-hour Sprint Retrospective**. Less mature teams try to shorten this to 1 hour or less, but it takes at least 1½ hours to truly dig into your team's tools, processes, structure, and culture and come up with a meaningful improvement, even for a week-long sprint.
- **Connect and celebrate** as a team after the retrospective. *This is not an official Scrum event*, but it's very helpful to spend quality, unstructured time hanging out with colleagues.

Sprint Planning

You can find a brief video about *Sprint Planning* at ap.tips/planning.

Goals

- Identify a specific, measurable, achievable *Sprint Goal* that can be accomplished in the *Sprint*.
- Realistically forecast what *Increment* can be delivered in the upcoming *Sprint*.
- Jointly plan how the *Increment* will be achieved.

Timing

- Typically 1–2 hours per week of *Sprint* duration, up to a maximum of 8 hours for a one-month *Sprint*.
- If you wonder why this is a good use of your limited developer time, see ap.tips/bottlenecks.
- Occurs at the beginning. The conclusion of *Sprint Planning* officially starts the *Sprint*.

Participants

- **Required:** The entire *Scrum Team*.
- **Invited:** Stakeholders and other subject matter experts as selected by the team to support the *Scrum Team* as they see fit. Participants who are not *Scrum Team* members must understand that this is not their event and they may not interfere with the team's process, only support the team.

Inputs

- A healthy *Product Backlog* (ap.tips/healthybacklog) with 1½–2 sprints worth of prioritized *Backlog Items* that are mostly "*Ready*" (ap.tips/dor) to work on.
    - *Backlog Items* include User Stories (ap.tips/userstory), defects, spikes, chores, etc. Anything the *Development Team* wants to work on should be characterized as a *Backlog Item* on the *Scrum Team's* single *Product Backlog*.
- The latest product *Increment* that will be built upon.
- The *Development Team's* likely *Capacity* during the *Sprint,* accounting for expected leave.
- The *Development Team's* historical performance. Many teams call this Velocity and measure it in Story Points, but note that this is not officially part of the Scrum framework.

Outputs

- A prioritized *Sprint Backlog* that the team expects—ideally based on data and not just wishful thinking—to fully complete by the end of the *Sprint* time-box.
- An initial sense of how the *Development Team* (ap.tips/devteam) will work together to accomplish the *Sprint Goal* and deliver the product *Increment*.
- A *Sprint Goal* that can be met through the implementation of at least most of the *Sprint Backlog*. It should be possible to meet the *Sprint Goal* without completing the entire *Sprint Backlog*.

Sample 3-Hour Sprint Planning Agenda (Two-Week Sprint)

- **5 minutes:** Check in with everyone socially, especially if you are a distributed team.
- **5 minutes:** If your team hasn't done so already, review incomplete *Backlog Items* from the last Sprint and decide what to do with them. Do not reestimate the remaining work. No partial points should be credited to the prior sprint for incomplete work.
- **2 minutes:** Identify your *Capacity* for the sprint. Each *Development Team* member should indicate what days they expect to be unavailable. State your *Capacity* in days without attempting to account for Scrum events or meetings. For instance, a 5-member *Development Team* conducting two-week sprints has 50 days of capacity at full strength and a capacity of 47 if three team members each take a day off. Don't count the *Product Owner* (ap.tips/po) or *Scrum Master* (ap.tips/sm).
- **3 minutes:** Calculate your estimated Velocity based on the Development Team's historical performance. All other things equal, we use the average of the last 3–4 sprint's *Capacity*-adjusted Velocity adjusted for current and forecasted *Capacity*:

  $$Forecast\ Velocity = \frac{\left(\frac{[Sprint\ n-3's\ Velocity]}{[Sprint\ n-3's\ Capacity]}\right) + \left(\frac{[Sprint\ n-2's\ Velocity]}{[Sprint\ n-2's\ Capacity]}\right) + \left(\frac{[Sprint\ n-1's\ Velocity]}{[Sprint\ n-1's\ Capacity]}\right)}{3} \times [Sprint\ n's\ forecast\ capacity]$$

  For instance, if you completed 25, 24, and 31 Story Points over the last three sprint with 45, 43, and 50 days of *Capacity* respectively and you have a forecasted *Capacity* of 47 this sprint, your forecasted Velocity for this sprint is $\frac{\left(\frac{25}{45}\right) + \left(\frac{24}{43}\right) + \left(\frac{31}{50}\right)}{3} \times 47 \approx 30$.

  A few Scrum-oriented tools on the market will calculate a forecasted Velocity for you, but we are not aware of any that address all edge cases well, particularly accounting for changes in *Capacity* and the number of *Development Team* members. It's probably easiest to use the free Velocity calculator we've created for your use, located at ap.tips/velocitytool.

  If the work in process in your carried-over *Sprint Backlog* meets or exceeds your forecast velocity and it is all still valuable to work on, we strongly recommend against taking in *any* new work this *Sprint*. You may even need to *remove* some items. This can be painful, but it's honest empirical process control (ap.tips/empirical) : the data available to us tells us that this is all that our *Development Team* is capable of, even if we wished we could do more.
- **5 minutes:** The *Product Owner* and team should agree on a *Sprint Goal* and prepare to select *Backlog Items* that will achieve the *Sprint Goal*.
- **70 minutes**: Determine what can be done this *Sprint*. Review the *Backlog Items* at the top of your *Product Backlog*. Discuss each *Backlog Item* briefly as a team before adding it to your proforma *Sprint Backlog*. If a new *Backlog Item* has appeared since you've last gotten together as a team for *Backlog Refinement*, discuss the item until the team is relatively satisfied that they understand what it is they are being asked to build, as well as the value it brings to the end user. Stop adding items to your *Sprint Backlog* when the *Development Team*—not the *Product Owner* or *Scrum Master*— has concluded that they have the right amount of work for the *Sprint* timebox. The *Product Owner* and *Development Team* should renegotiate *Backlog Items* as necessary.
  - To the extent possible, capture the outcome from the customer's perspective, not the behavior of the system. For instance, a good *Backlog Item* title might be "As a coffee drinker, I can have coffee that's just the right temperature so that I can delight in its

ideal taste," not "Integrate the Acme steam-heating element into the brew assembly." A good *Backlog Item* description is "the served coffee is between 60–70°C (140–155°F)," not "The brew-port Type-K thermocouple shall read between 2.436 and 2.644mV at dispensing." Give room for your *Development Team* to leverage their expertise and figure out the how. (Agile principles #1 and 11)
    - Try to have at least six *Backlog Items* per sprint. Otherwise they may be too large.
    - Although not officially part of Scrum, many teams use a Definition of "*Ready*" to reduce the risk that they will pull an item into their sprint that is not "*Ready*" to work on yet.
    - If your team uses Story Points (ap.tips/storypoints), this is the last responsible moment to estimate.
- **15 minutes:** Break. Leave the room, walk around, and take care of your individual needs.
- **90 minutes:** Determine how the chosen work will be completed. The *Development Team* completes a high-level design of the systems and engineering approach to get the selected *Backlog Items* to "*Done*." (ap.tips/dod) Note that you do not have a *Sprint Backlog* until you have identified both the selected *Backlog Items* and your approach for completing them.
    - This is often done by "tasking out" *Backlog Items* into several smaller tasks. These tasks are often described as brief implementation steps that take a day or less to complete. For example, a simple *Backlog Item* might be "As a new visitor to the site I can register for a deals newsletter so that I save money on my next purchase," and the tasks might be "design form visually," "write HTML," "write CSS", "write JavaScript form validation", "write server side form validation", "add customer information to marketing database", "write unit tests", "write integration tests", "smoke-test in staging," and "deploy to production."
    - The *Development Team* can divide and conquer this step as they see fit. It's best to complete most of this breakdown at *Sprint Planning*, especially for items that will be worked on first.
    - If this effort causes the *Development Team* to doubt the ability to reach the *Sprint Goal* and deliver the *Sprint Backlog* by the end of the *Sprint* time-box, they should renegotiate the *Backlog Items* with the *Product Owner*.

Tips

- If your team does not have a known, stable Velocity (ideally 3-4 sprints worth of data) or chooses not to use a quantitative approach, try using "Fists of Five" (ap.tips/fistsoffive).
- The *Development Team* cannot and must not make a firm commitment to delivering the entire *Sprint Backlog* or even the *Sprint Goal*. (ap.tips/tradeoffs) Attempting to do so will only lead to burnout, poor quality, loss of trust, and/or attempts at "gaming the system." Instead, the *Sprint Backlog* and *Sprint Goal* is a focusing tool that limits context switching and a good-faith forecast that the *Development Team* attempts to live up to.
- A mature team should be able to deliver 85-115% of the planned work or Velocity every sprint. If a team consistently underdelivers, this usually means that they are not being truly empirical and they are taking on more work than their historical performance supports.
- A team with a new backlog should spend close to 10% of their time building out their backlog *before* each *Sprint Planning* session. A well-established team can afford to spend less. If you are finding that it is hard to fully complete the "what are we building" phase of *Sprint Planning* within a one-hour per Sprint week time-box, you likely need more preparation from the *Product Owner* and/or *Backlog Refinement*.

Daily Scrum

Also known as “Standup.”
You can find a brief video about the *Daily Scrum* at ap.tips/dailyscrum.

Goals

- Inspect progress towards the *Sprint Goal* and completion of the *Sprint Backlog*.
- Plan work for the next 24 hours and renegotiate the *Sprint Backlog* if necessary.

Timing

- Never more than 15 minutes.
- Two or more of the participants may *optionally* stick around after the *Daily Scrum* for more detailed discussion, blocker resolution, replanning, and/or adaptations.

Participants

- **Required:** The entire *Development Team* (ap.tips/devteam).
- **Strongly suggested:** The *Product Owner* (ap.tips/po) and *Scrum Master* (ap.tips/sm).
- **Invited:** Stakeholders and other subject matter experts may always attend. Participants who are not *Scrum Team* members must understand that this is not their event and they may not interfere with the team’s process, only support the team.

Inputs

- The *Sprint Goal* and *Sprint Backlog*
- Optional: useful forecasting tools such as a Burn-Down Chart (http://ap.tips/burndown)

Output

- A plan for the next 24 hours and a newly updated *Sprint Backlog* as necessary.

Sample 15-Minute Daily Scrum Agenda

- **13 minutes:** Each *Development Team* member answers (from the Scrum Guide™, scrumguides.org):
    - What did I do yesterday that helped the *Development Team* meet the *Sprint Goal*?
    - What will I do today to help the *Development Team* meet the *Sprint Goal*?
    - Do I see any impediment that prevents me or the *Development Team* from meeting the *Sprint Goal*?
- **2 minutes:** The *Scrum Team* updates and inspects the state of the *Sprint Backlog* and Burn-Down Chart (ap.tips/burndown).
- **Optional, following the event:** People can stick around and discuss “Parking Lot” topics.

Tips

- Anyone, including those outside the team, may raise “Parking Lot” topics. Further, if any *Scrum Team* member feels that a discussion during the *Daily Scrum* is taking too long and threatens the timebox, they can suggest further discussion be held for the “Parking Lot.” Anyone who wishes to do so can stick around after the *Daily Scrum* and discuss these topics. Participation is fully voluntary at this point. Examples:
    - A developer would like to briefly engage in a deeper face-to-face discussion with a few other *Scrum Team* members that does not warrant another meeting.
    - A developer has discovered an unexpected edge case and would like to strategize on a resolution with the rest of the *Scrum Team*.
    - The *Scrum Team* realizes that it’s *Sprint Goal* is at risk and they wish to adapt their *Sprint Backlog* accordingly.
    - A stakeholder has questions or concerns they’d like to bring to the *Scrum Team*.
- Conduct the *Daily Scrum* at the same place and time every day. Some teams will cancel the *Daily Scrum* on the last day of the sprint, but we recommend against that as the team could use this to coordinate their final effort against the *Sprint Goal*.
- Distributed teams should consider scheduling more than 15 minutes for this event. The *Daily Scrum itself* is still time-boxed for 15 minutes, but it can be very useful to keep the video conferencing bridge reserved for “Parking Lot” topics. Distributed *Scrum Team* members are less likely to speak to one another informally throughout the rest of the day.
- It doesn’t matter how the *Development Team* conducts their *Daily Scrum* so long as it remains within the 15-minute timebox and meets the same goals. For instance, some teams prefer informal discussion over a standard set of questions.
- The *Daily Scrum* is not a status review or report for anyone, including *Product Owners*, *Scrum Masters*, or (especially) stakeholders. Don’t let it become one.
- Start the Daily Scrum on time, even if much of the team is missing… within a few days, attendance should improve as being late to a meeting in progress is more uncomfortable than being late to a meeting that hasn’t started yet.
- As the member of the *Development Team* is speaking, watch out for the following:
    - Is everyone else looking directly at the person speaking and paying full attention? It can be a fun game for the speaker to look for anyone who isn’t paying full attention and nominate them as the next speaker.
    - Are they speaking directly to work from the *Sprint Backlog*, or are they working on something else? Any effort towards work not in the *Sprint Backlog* potentially impacts the *Product Owner’s* ability to support the team and saps the team of velocity made good against the *Sprint Goal*.
- It’s not the *Scrum Master’s* job to always facilitate the *Daily Scrum*, only to ensure that it happens smoothly and within the 15-minute time-box.
- Asking able-bodied participants to remain standing throughout the entire event—hence the alternative name “Standup”—helps keep the meeting short because it becomes physically uncomfortable for most participants for it to run longer than 15 minutes.

Sprint Review

You can find a brief video about *Sprint Review* at ap.tips/review.

Goals

- Elicit feedback on the *Increment* from stakeholders—ideally even end users—and collaboratively improve the *Product Backlog* together.
- Foster a culture of transparency and trust between the *Scrum Team* and stakeholders.

Timing

- Up to 4 hours for a one-month *Sprint*. Most commonly, 15-60 minutes per week of *Sprint* duration. Ideally, 1 hour for a 2-week *Sprint*.

Participants

- **Required:** The *Scrum Team*
- **Strongly Suggested:** Key stakeholders and end users invited by the *Product Owner* (ap.tips/po).

Input

- The updated *Sprint Backlog* and knowledge as to whether or not the *Sprint Goal* was met.

Output

- An updated *Product Backlog* that reflects stakeholder feedback and new knowledge about the product's context, such as the competitive landscape and the company's resources.

Sample 1-Hour Sprint Review Agenda

- **5 minutes:** The *Product Owner* welcomes the stakeholders, indicates whether or not the *Sprint Goal* was met, and indicates which *Sprint Backlog* items are and are not "*Done*."
- **5 minutes:** The *Development Team* (ap.tips/devteam) briefly discusses the highs, lows, and blockers encountered during the *Sprint* and how they were responded to.
- **30 minutes:** The *Development Team* demonstrates the "*Done*" *Backlog Items*, solicits honest and critical feedback, and answers questions.
- **5 minutes:** The *Product Owner* discusses the current state of the *Product Backlog*. They can also offer forecasts, such as the likelihood of completing a given scope by a given date, the expected scope for a given date, or the expected date for a given scope.
- **5 minutes:** Stakeholders share new knowledge about how the market or potential uses for the product may have changed during the *Sprint* to help the *Scrum Team* make more informed choices.
- **5 minutes:** Review the timing, resources, potential features, and distribution channels for the next potential release of the product.

- **5 minutes:** Collaboratively decide what to do next so as to better prepare the *Product Backlog* for the coming *Sprint Planning*.

Tips

- This is *not* a status meeting. Rather, it is an opportunity for stakeholders to collaborate with the team and improve the *Product Backlog* with shared context and learnings.
- Don't demonstrate work that isn't "*Done*." It can significantly erode trust between the *Scrum Team*, stakeholders, and end users if the demonstrated work doesn't get shipped as expected. It also allows the *Scrum Team* to persist the illusion that incomplete work is still valuable. Remember: work in process is a liability, not an asset. (ap.tips/toc)

Sprint Retrospective

You can find a brief video about *Sprint Retrospective* at ap.tips/retro.

Goal

- The *Scrum Team* should sincerely self-reflect on their tools, process, structure, and culture and propose improvements or experiments that will be attempted in the following *Sprint*.

Timing

- Up to 3 hours for a one-month sprint. Typically 1½ hours for a 2-week sprint.
- Occurs after the *Sprint Review* so that observations at this event can also be learned from.
- Occurs just prior to *Sprint Planning* so as to minimize the gap between *Sprints*.

Participants

- The entire *Scrum Team*. The *Product Owner* and *Scrum Master* should participate as peers to the *Development Team*.
- No other may attend unless specifically invited to do so by the entire *Scrum Team*.

Input

- Objective data, information, and observations addressing:
    - The *Scrum Team's* progress against the *Sprint Goal* and *Sprint Backlog.*
    - The well-being of the people and relationships in and around the *Scrum Team.*
    - The effectiveness of the Scrum Team's tools, process, structure, and culture.
- A culture of courage, commitment, respect, and openness.

Output

- *Backlog Items* capturing improvement experiments to be attempted in the next *Sprint*.
- Changes to the *Definition of "Done"* (ap.tips/dod)
- Unofficially, changes to the Definition of "*Ready*" (ap.tips/dor) and Working Agreement (ap.tips/workingagreement).

Sample 1½-Hour Sprint Retrospective Agenda

- **5 minutes:** "Set the stage." This is a good chance to break the ice with your team and remind ourselves why this is so important.
    - Invite each team member to share one-word or a brief phrase that describes how they're feeling.
    - Have a team member read Norm Kerth's "Retrospective Prime Directive" to the rest of the team: "Regardless of what we discover, we understand and truly believe that everyone did the best job they could, given what they knew at the time, their skills and abilities, the resources available, and the situation at hand."

    - Invite each team member to share specific appreciation for one another. For instance, one team member might say, “Alan, I really appreciate how you helped me figure my way out of the memory leak issue when you reviewed my pull request.”
- **5 minutes:** “Gather data.” What happened in the sprint? I like to use a “positives / deltas / insights” collection device. Ask each team member to come up with as many positives, deltas or things they’d like to change, and insights as possible. Have them write each positive, delta, and insight individually on sticky notes. When the team has run out of thoughts or the timebox has elapsed, place the sticky notes on the wall and group like items together.
- **3 minutes:** Read and deduplicate your topics.
- **2 minutes:** Vote on topics. The highest-voted topics will be discussed. You can give everyone 2 or 3 votes that they can distribute any way they like. If you’d like to get fancy, give people the approximate square root of the topic count in votes (~9 topics = 3 votes, ~16 topics = 4 votes, ~25 topics = 5 votes, etc).
- **70 minutes:** For each of the top 3 voted topics, spend 15-20 minutes:
    - “Generating insights.” If a positive reads “The team is doing a good job of collaborating with one another” and a delta reads “Our staging server keeps going down,” try to understand the root cause of these conditions. Some teams find it helpful to ask “why” five times (give or take) until you get to your root cause. The book “Agile Retrospectives” (see Tips) suggests other exercises as well. If you’re rarely surprised by your root causes, you may not be digging deep enough.
    - “Actions.” I like to expand this into Decisions and Actions. For each item, discuss as a team suggestions about how you can persist and deepen the positive, resolve or mitigate the delta, and capitalize on the insight. You’ll find that your suggestions fall naturally into two buckets: decisions, which are changes to the *Definition of “Done”*, Definition of “*Ready*”, Working Agreement, and actions, which are simply new *Backlog Items*. For example:
        - The team is tired of events starting late due to tardiness. They could vote to add a rule to their Working Agreement that imposes a penalty for being late.
        - The team discovers that their staging server needs several hours of maintenance. They could decide to add a *Backlog Item* to the *Product Backlog* that reads, “As a developer, I have a more reliable staging server so that I can develop software more effectively.”
    - Stay focused: it’s better to have one or two improvement items for the next *Sprint* than to have too much to change at once.
- **5 minutes:** “Wrap up.” If you haven’t done a round of appreciations yet, this is a nice time. Spend a few minutes retrospecting on the retrospective. How did it go for the team? What would they like to try next time?

Tips

- If your team is using Kanban, try running a *Sprint Retrospective* every two weeks.
- If your team is using 1-week *Sprints* and complaining about the number events, you could try running a *Sprint Retrospective* every two weeks before switching to two week sprints. While not as ideal, it’s still useful for new teams to have weekly *Sprint Planning* and *Sprint Review* events. You might as well find a middle ground.

- Use 3M's "Super Sticky Post-It®" notes (amzn.to/2JDaBhE), not the generic sticky notes you find in many office supply cabinets. They really do stick better, and the last thing you want during your *Retrospective* is sticky notes falling off the walls.
- Good digital equivalents to sticky notes are Ideaboardz.com (free but doesn't work in Safari and a little buggy), FunRetro.io (paid), and Retrium.com (paid).
- Read "Agile Retrospectives: Making Good Teams Great" by Esther Derby and Diana Larsen (amzn.to/2V12lx9) and/or "Project Retrospectives: A Handbook for Team Reviews" by Norm Kerth. (amzn.to/346p7YI) . The sample Sprint Retrospective agenda is based on Esther and Diana's book, while the "Retrospective Prime Directive" comes from Norm's book.
- Bring snacks! If you're working remotely, have a treat together anyway.
- Check out ap.tips/primedirective for a thought-provoking discussion around the Retrospective Prime Directive.
- The *Scrum Team* should explicitly use either the "Vegas Rule" ("Whatever happens in Vegas, stays in Vegas", meaning nothing is discussed about the retro without unanimous explicit consent) or "Chatham House Rule" ("When a meeting, or part thereof, is held under the Chatham House Rule, participants are free to use the information received, but neither the identity nor the affiliation of the speaker(s), nor that of any other participant, may be revealed."). These rules help create the necessary safe container for the team to bring up difficult issues.

Backlog Refinement

You can find a brief video about *Backlog Refinement* at ap.tips/refinement.
Note: this is *not* an official Scrum event.

Goals

- Detail, improve, estimate, and prioritize *Product Backlog* items collaboratively to increase shared understanding, reduce risk, and shorten the duration of *Sprint Planning*.
- 1½ to 2 sprints worth of "*Ready*" *Backlog Items*.

Timing

- Up to 10% of the total *Sprint* duration. Many *Scrum Teams* specifically schedule one or more *Backlog Refinement* events that are held during the sprint. We like to schedule at least 1½ hours of *Backlog Refinement* per week.
- If you wonder why this is a good use of your limited developer time, see ap.tips/bottlenecks.

Participants

- **Required:** The *Product Owner* and at least one member of the *Development Team*.
- **Strongly Suggested:** The entire *Scrum Team* and any useful subject matter experts or stakeholders. Even if a developer does not have enough context to immediately contribute to improving the *Backlog Items*, they can increase their understanding of the domain and identify novel approaches, edge cases, or risks that others on the team may have missed.

Input

- A *Product Backlog*, ideally with *Backlog Items* that have already received a solid attempt at elaboration from the person who created them.

Output

- An healthy, freshly prioritized, and often estimated *Product Backlog* (ap.tips/healthybacklog).

Sample 1½-hour Backlog Refinement Agenda

- **90 minutes:** Refine the *Product Backlog* together to make more stories "*Ready*." Start at the top of the *Product Backlog*. For each *Backlog Item*:
    - hold a brief conversation about the description, order, and value
    - Make big Backlog Items into smaller ones using Vertical Slicing (ap.tips/slicing)
    - capture the conversation in the *Backlog Item*'s description (ideally as a bulleted-list of specific, testable conditions we call "Acceptance Criteria")
    - have the *Development Team* estimate the Backlog Item in story points (ap.tips/storypoints) using Planning Poker (ap.tips/planningpoker) or Affinity Estimating (ap.tips/affinityestimating).

    - allow the *Scrum Team* to propose changes to the order based on a thoughtful consideration of its value, estimate, and dependencies (the *Product Owner* makes the final decision

Tips

- Most of your Backlog Items should be characterized as User Stories (ap.tips/userstory).
- The *Product Owner* should send out a list of the *Backlog Items* to be discussed before the *Backlog Refinement* event. This allows the *Scrum Team* to read ahead and familiarize themselves with their description, order, and value. It also helps communicate the full scope of the *Backlog Refinement* event to encourage brevity. If the *Scrum Team* gets these stories to "*Ready*" early it can end the event early.
- Have the *Scrum Master* keep track of how long the team is spending discussing a given *Backlog Item* and keep the team aware of the passage of time to help encourage brevity. They should say something like "We've been talking about this *Backlog Item* for 10 minutes, should we continue or park it until Pat can get us more details?" If you find yourselves talking about a story for too long, it may be that more preparation or research is required and the *Backlog Item's* author should delay discussion for the time being if possible.
- Read the books "User Story Mapping: Discover the Whole Story, Build the Right Product " by Jeff Patton (amzn.to/2JF29yi) and "User Stories Applied: For Agile Software Development" by Mike Cohn (amzn.to/3dVL692).
- Teams with new backlogs should strongly consider 3-4 hours of *Backlog Refinement* per week to create a high-level sense of the team's longer-term vision.
- Don't do detailed planning too far in the future. Scrum is an empirical, iterative approach to product development. This means that you must continuously update and improve your Product Backlog based on what you learn from stakeholders and end users with each *Sprint Review* and product *Increment* release. If you plan too far in the future, you run the risk of wasting time trying to plan without enough real-world feedback and becoming too attached to a poorly-informed strategy.
- Use Planning Poker when estimating a few *Backlog Items* during *Backlog Refinement* or *Sprint Planning*. Use Affinity Estimating when you have many *Backlog Items* to estimate.
- Take your time when estimating *Backlog Items*. Estimation serves a dual purpose. Yes, it gives the team a reasonably inexpensive quantitative measure of their historical performance and a reasonable evidence-based forecast. However, we can also learn a great deal from the ensuing conversation. Suppose one developer rates a *Backlog Item* as a "1," while another rates it as a "5" (five times as much complexity, uncertainty, and/or effort). If the *Scrum Team* accepts an average or the most senior developer's opinion, they may miss out on an opportunity for shared knowledge. Perhaps the developer who estimated the Backlog Item a "1" is aware of a faster technique. Perhaps the other is aware of a challenging edge case. Take the time to talk to one another. A few minutes here could save hours later.
- Estimate all *Backlog Items*, even defects. Some *Development Teams* don't like to estimate defects because the root cause can sometimes be difficult to predict. The advantage of making the attempt is that your team will be better able to maintain a smooth, steady flow of work when the expected cost of resolving defects is also accounted for.